BOG 9: Productivity, Composition, and Interoperability

ASCR Workshop on Extreme Heterogeneity in HPC

24 Jan 2018

BOG 9 Contributors

Moderator(s): Bob Lucas (USC), David Bernholdt (ORNL), and Katie Schuman (ORNL)

Additional BOGists:

- Rob Falgout (LLNL)
- John Shalf (LBNL)
- Anshu Dubey (ANL)
- Ron Brightwell (SNL)
- EJ Park (LANL)
- Bruce Jacob (Maryland)
- Lucy Nowell (DOE)
- Sherry Li (LBNL)
- Rich Carlson (DOE)
- Andrew Younge
- Kevin Pedretti (SNL)
- Chris Rossbach (Texas)
- John Wu (LBNL)
- Esam El-Araby (KU)
- Ewa Deelman (USC)

BOG 9 Capability Targets for Extreme Heterogeneity

BOG 9 brainstorming and discussion of capabilities that will be needed in the 2025-2035 timeframe to make increasingly heterogeneous hardware technologies useful and productive for science applications.

BOG 9 Targets for 2030

Composition: using multiple codes

- perhaps simply reuse
- composed together for multi-physics capabilities
- independent applications, cooperating in a workflow

Productivity:

- Maximizing discovery per dollar (or Watt, or investigator)
- Maximizing productivity of scientific developers
- Maximizing throughput of DOE computing ensemble

Interoperability:

- Interoperability of multiple, extremely heterogeneous (EH) systems
- Code portability across multiple, EH systems

BOG 9 Current Status

Heterogeneity is already a problem today

Node level:

- Various sizes and types of memory (DDR, GDR, HBM)
- Presence (or absence) of accelerators (Cell, KNL … GPU, APU, TPU, FPGA)

System level:

- Crop rotation
- swim lanes
- “workflow-scale” integration of multiple systems

It’ll rapidly get worse, witness the plenary talks (e.g., looming persistent main memory)

(Maybe several viewgraphs)

BOG 9 Current Capability

- Composition:
    - Labor-intensive process to merge into single code base
    - Frameworks (Sierra)
    - Runtime communication (Ron B. comment)
- Productivity:
    - Metrics for human productivity largely absent, other than anecdotal best practices
    - Performance engineering of code is largely labor intensive (in spite of early autotuning results)
- Interoperability:
    - Standard languages and libraries to facilitate moving existing codes amongst machines
    - Labor intensive process to adapt to constraints (e.g., small memory) or opportunities (e.g., new accelerators)

BOG 9 Challenge Assessment

Today, variation in architecture (e.g., AVX width changing) makes generations of what are nominally the same systems heterogeneous in time, requiring revisiting software design decisions. Varying the balance of memory and accelerators from one system to the next further complicates this.

Near term, this will be complicated by richer set of ISAs and the new memory technology expected to be deployed. Longer term, exploitation of SOC technology will likely lead to richer set of CMOS accelerators, and these will be complimented by accelerators exploiting alternative modes of computation such as neuromorphic and quantum. This will lead us progressively to a post-ECP, EH computing environment.

Enabling a broad range of computational scientists to effectively approach and exploit the capability of this EH environment will be extremely challenging. This is particularly true in science and engineering, given the large body of existing code, representing a huge investment of labor, and incorporating the expertise of rare, specially trained people.

As it stands today, where EH is overcome, it will likely be by significant investments in labor. Where that is not possible (cost or availability of competent people), the pace of discovery will languish, and opportunity will be lost. Research into the development of new abstractions and tools to make these people more effective, or ideally automate much of the labor, is the only credible path forward.

BOG 9: list of key research challenges

- How will can we map a large software system onto hardware in the most effective way?
  - Need to have cost models embedded in each software component, and they will need to be automatically generated (too complex otherwise)
  - Optimizing individual applications vs optimizing overall system throughput
- Mapping data representations between different types of accelerators (really rich space of representations)
  - How much can be automated?
- Programming abstractions to make hetero systems easier to program
  - Trade-off human labor vs compute effort, which will change over time
  - Interoperability and composition of abstractions, including new ones as EH components are deployed
- Correctness and reproducibility in an environment where different accelerators behave very differently
  - May sometimes need bitwise reproducibility (e.g., debugging code)
  - Mathematical challenge in general to understand appropriate definition of crrectness

BOG 9 Possible Research Directions Summary

- PRD 9.1 - Reinvent memory to exploit persistent technology
  - Opportunity to facilitate composition of codes, and add facilities like journaling
  - Opportunity to eliminate overheads like TLB by revisiting topics like protection
  - Opportunity to rethink DM H/W abstraction, and embrace in-situ storage (icl. C/R)
- PRD 9.2 - Develop hierarchies of abstraction to maximize human labor
  - Spanning DSLs to device level codes for ninjas (e.g., CUDA)
- PRD 9.n - Develop new runtime algorithms and tools to adapt in real time
  - Automatic generation of performance so one can reason about what is being observed
  - Adapt to runtime heterogeneity (freq. scaling) or better, manage it (light up dark Si)

PRD 9.n : Short title of possible research direction

- One paragraph description (3 sentence/bullet)
- Research challenges
  - Metrics for progress
- Potential research approaches and research directions
- How and when will success impact technology?

(Maybe several viewgraphs)